Slim Scissors: Segmenting Thin Object from Synthetic Background

Kunyang Han[1,2⋆], Jun Hao Liew[3], Jiashi Feng[3], Huawei Tian[4], Yao Zhao[1,2], and Yunchao Wei[1,2]

[1] Institute of Information Science, Beijing Jiaotong University
[2] Beijing Key Laboratory of Advanced Information Science and Network Technology
[3] ByteDance
[4] People's Public Security University of China

https://kunyanghan.github.io/SlimScissors/

Abstract. Existing interactive segmentation algorithms typically fail when segmenting objects with elongated thin structures (*e.g.*, bicycle spokes). Though some recent efforts attempt to address this challenge by introducing a new synthetic dataset and a three-stream network design, they suffer two limitations: 1) large performance gap when tested on real image domain; 2) still requiring extensive amounts of user interactions (clicks) if the thin structures are not well segmented. To solve them, we develop **Slim Scissors**, which enables quick extraction of elongated thin parts by simply brushing some coarse scribbles. Our core idea is to segment thin parts by learning to compare the original image to a synthesized background without thin structures. Our method is model-agnostic and seamlessly applicable to existing state-of-the-art interactive segmentation models. To further reduce the annotation burden, we devise a similarity detection module, which enables the model to automatically synthesize background for other similar thin structures from only one or two scribbles. Extensive experiments on COIFT, HRSOD and ThinObject-5K clearly demonstrate the superiority of Slim Scissors for thin object segmentation: it outperforms TOS-Net by 5.9% IoU_{thin} and 3.5% $\mathcal{F}$ score on the real dataset HRSOD.

Keywords: Interactive image segmentation; thin object segmentation

1 Introduction

Interactive image segmentation denotes the task of extracting the object-of-interest given some user-hints, such as clicks [44,24,23,48], scribbles [3,11,2,12], polygons [4,1,27] or bounding boxes [35,43,42]. This task has received much attention over the past few years due to its wide application domains ranging from image editing, medical image analysis [41] to dataset annotation [4]. The segmentation process usually iterates between: (i) user providing feedback based

⋆ Work done during an internship at ByteDance.

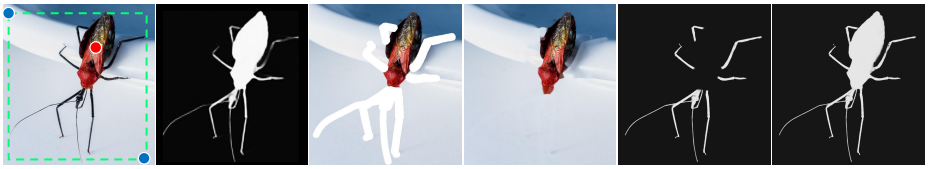


Fig. 1: **Overall idea.** Our Slim Scissors first takes the inside-outside guidance (IOG) [48] (a) to produce a coarse segmentation outlining the main body (b). Given some *coarse* scribbles around the thin structures (c), we construct a synthetic background image without the presence of thin parts (*e.g.*, bug legs and antennas) via image inpainting algorithm (d) and feed such (image, background) pair to a network for segmentation of thin parts (e). Finally, both features are fused to produce the final segmentation output (f).

on the errors of current segmentation, and (ii) the segmentation model updating its prediction accordingly; and terminates when the user is satisfied with the final segmentation result.

Existing deep learning-based interactive segmentation algorithms have shown exceptional performance in this task by typically requiring only a few clicks to obtain a high-quality segmentation mask [44,43,1,30,48,26]. For example, the current state-of-the-art Inside-Outside Guidance (IOG) [48] only needs three clicks to attain >90% IoU accuracy. However, despite the overall good performance, these state-of-the-art methods can hardly be deployed to segment objects with elongated thin parts (*e.g.* bug legs and bicycle spokes), which is the main focus of this work. As a result, excessive user interactions for correcting those missing or wrongly segmented thin parts becomes unavoidable.

Recently, Liew *et al.* [22] found out that the failure in segmenting thin objects can be mainly attributed to two factors: (i) coarsely annotated dataset used for training interactive segmentation models (*e.g.*, PASCAL [10] and COCO [25]) and (ii) imbalanced distribution between thin and non-thin pixels, which cause difficulty in learning using standard pixel-wise cross-entropy loss. To tackle these challenges, they collected a large-scale synthetic dataset specifically tailored for segmentation of elongated thin objects, called ThinObject-5K. Based on this, they proposed a three-stream network named TOS-Net and demonstrated significant performance improvement over the baselines.

Despite the outstanding performance of TOS-Net [22] in segmenting thin structures, it still exhibits the following drawbacks: (i) severe performance degradation when testing on real images due to the large domain gap between the synthetic training samples and real testing images (*e.g.* ∼10% and ∼20% IoU_{thin} [22] on COIFT [31] and HRSOD [47] dataset, respectively); (ii) since TOS-Net follows the extreme clicking paradigm of DEXTR [30], it typically requires extensive user inputs (boundary clicks) for correction, especially when it comes to refining elongated thin parts, offering an unsatisfactory user experience.

To tackle this issue, we develop a new interactive segmentation paradigm, called **Slim Scissors** that enables fast extraction of elongated thin parts by simply drawing some *coarse* scribbles to cover them. Note that we employ *thick/ coarse* scribbles whose radius is much larger than that of the thin structures (Fig. 1 and 6). As a result, scribbling along thin structures is fast. Compared to boundary clicking, coarse scribbling is not only much more convenient as users can quickly brush over the thin parts instead of carefully clicking on them, it also contains significantly richer information (*e.g.* shape prior) than sparse clicks.

The scribbles are then used to construct a synthetic background simulating the absence of thin parts. The underlying motivation is that the contrastive information in such an (image, background) pair can help enhance the signals of elongated thin parts, simplifying the learning process. For instance, segmenting thin spokes from a bicycle wheel becomes easier if able to compare the original image of bicycle wheel with a background image without the thin spokes. Based on this, we train a network that learns to segment thin objects based on such (image, background) pair input. Such design can be easily augmented with a generic interactive segmentation architecture to form an end-to-end trainable network. As shown in the experimental section, this simple design works surprisingly well.

As compared to boundary clicks, drawing *coarse* scribbles doubtlessly reduces the annotation burden by a significant margin. To further reduce the annotation cost, we propose a similarity detection module that automatically finds others similar thin structures guided by the current drawn scribbles. Taking Fig. 1 as an example, one only needs to coarsely draw one or two scribbles on the bug's legs while our similarity detection module generates coarse scribbles covering the remaining legs by computing per-pixel similarity with the user-provided ones. Such module can significantly reduce human efforts in drawing scribbles when many similar thin patterns appear in the same image.

Overall, the key contributions are as follows: 1) We develop **Slim Scissors**, a simple and effective tool that enables users to efficiently and accurately acquire elongated thin parts by simply drawing some coarse scribbles around them. The success of Slim Scissors can be attributed to the novel idea of segmenting from synthetic background. 2) We propose a similarity detection module to augment the Slim Scissors. This module significantly reduces the annotation burden when many similar thin patterns independently appear within the same image. 3) We perform extensive experiments on three publicly available benchmarks and show significant performance improvement upon the state-of-the-art on the two real datasets, COIFT [31] and HRSOD [47].

2 Related Works

Interactive Object Segmentation. Recently proposed interactive image segmentation algorithms are primarily driven by deep learning-based methods where significant improvements upon the traditional approaches have been demonstrated. Among them, Xu *et al.* [44] was the first to apply deep learning to this task. They proposed to apply Euclidean distance transformation to the

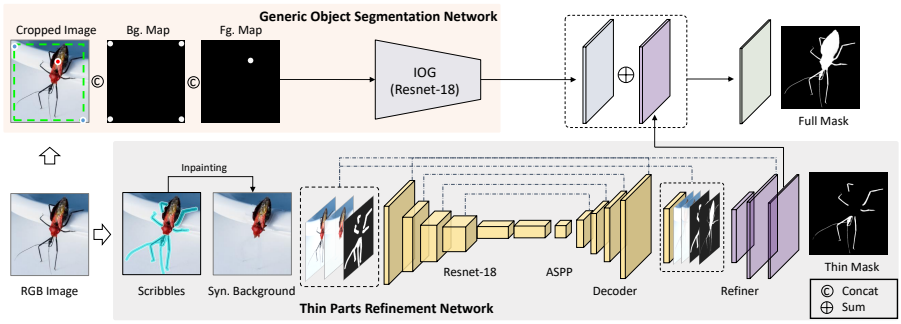


Fig. 2: **Slim Scissors.** Our segmentation network is composed of two parallel networks: (top) a Generic Object Segmentation (GOS) network that takes Inside-Outside Guidance (IOG) [48] as input to extract a coarse mask delineating the object main body; (bottom) a Thin Parts Refinement (TPR) network that first synthesizes a background without thin parts based on *coarsely* drawn scribbles. By comparing the (image, background) pair, our TPR networks produces a coarse prediction of thin parts, which is subsequently refined with a refiner to produce a mask of elongated thin structures only. Finally, features from both networks are fused to obtain the final segmentation mask. Note that we did not visualize the coarse mask prediction by IOG here for simplicity.

user clicks and train a FCN based on such (image, clicks) pair. Since then, various forms of inputs representation [41,29], interaction types [30,20,1,27,8] have been explored. Some works [24,15,26,7,13] seek to better exploit useful cues from user-provided inputs while others [21,23] tackles the ambiguity in interactive segmentation given limited user hints. More recent works such as [16,37,19] focus on inference-time optimization to improve segmentation quality. Despite their overall good performance in segmenting general objects, these approaches typically fail when deployed to segment objects with elongated thin parts.

Interactive Thin Object Segmentation. Vicente *et al.* [39] tackled the shrinking bias problem in graph cut when segmenting elongated thin objects by proposing a connectivity prior. Jegalka and Bilmes [17] developed cooperative cuts that favors homogeneous boundaries by penalizing the number of types of label discontinuities. Mansilla *et al.* [31] introduced connectivity constraint on Oriented Image Foresting Transform (OIFT) and demonstrated considerable improvement on segmentation of thin objects. Dong *et al.* [9] proposed a sub-Markov random walk (subRW) algorithm and showed that segmentation of thin objects can be handled by adding label prior to subRW. More recently, Liew *et al.* [22] made the first attempt to extend the deep learning-based interactive segmentation methods to handle elongated thin objects. Nevertheless, this approach performs poorly on real image domain due to the large domain gap with the synthetic training data. Furthermore, the boundary clicking paradigm offers a poor user experience, making their methods less useful for practical ap-

plication. In this work, we advocate the use of *coarse* scribble in replacement of boundary clicking to enable faster interaction and a novel idea of segmenting from synthetic background for more effective segmentation of thin structures.

3 Motivation

At the core of our approach is the novel idea of learning to segment thin parts from synthetic background where these backgrounds are synthesized based on the *coarsely* drawn scribbles highlighting the elongated thin structures. We begin by explaining the motivation before proceeding to explaining our approach.

Why choose scribbles for interaction? Boundary clicking adopted by [22], in general, serve as an efficient means for interaction when conducting interactive segmentation of general objects. However, we consider that boundary clicking may not be an optimal choice for selecting thin objects as precise clicking on the thin parts can be extremely time-consuming. For example, as shown in Fig. 1(a), when segmenting the beetle's antennas, users need to be extremely careful when clicking such that the clicks are correctly located on the boundaries of the antennas, offering an overall poor user experience. Differently, taking *coarsely* drawn scribbles to cover elongated thin parts is arguably faster and much more convenient, as users only need to roughly brush over them. As shown in Fig. 1 and 6, the scribble size is much larger than that of the thin structure, easing the annotation process. This motivates us to employ coarse scribbles for selecting the interested thin parts for segmentation. It is also worth mentioning that there exist a large body of works which employ scribbles for interactive segmentation [3,11,2,12,5]. However, our approach completely differs from prior works in terms of both motivation and fashion. Specifically, previous works typically draw scribbles *within* the object to indicate the target-of-interest whereas we take *coarsely* drawn scribbles to cover elongated thin parts. An alternative to coarse scribbling is patch annotation over image grid [8]. However, we argue that patch annotation over fix-sized grid is less flexible.

Why do we need synthetic background? There are two reasons why synthetic background will be beneficial for thin object segmentation. First, the challenge of thin object segmentation task mainly lies in that many thin parts are a few pixels wide, making their features be easily overlaid by around background after several downsampling operations, which are common settings in popular CNN backbones. Intuitively, if we are able to acquire the background under the thin parts, the contrastive information will help lift the signals of the inconsistent regions between foreground and background and relieve the negative effect caused by the downsampling operations. Second, thin parts are often with rich contextual background, making the background of thin parts be easily and accurately synthesized even with very simple inpainting algorithms.

Fig. 3: **Examples of generated synthetic background images** using `cv2.inpaint`. We challenge the reader to identify the missing thin structures by simply comparing the (image, background) pair [5].

4 Slim Scissors

We propose Slim Scissors, a novel, generic, and effective solution for interactive thin object segmentation. As shown in Fig. 2, the overall network consists of two parallel subnetworks, *i.e.*, (i) a Generic Object Segmentation (GOS) network for obtaining a coarse segmentation mask delineating the object's main body; (ii) a Thin Parts Refinement (TPR) network for extracting the elongated thin structures based on the user-provided scribbles. To obtain the final segmentation output, we simply element-wise sum the features prior to the last layer of both subnetworks and pass to a 1×1 `conv` with sigmoid activation. Explicitly disentangling the processing of thin parts from the main body in this way not only encourages each subnetwork to better focus on different aspect of the object (main body *vs*. elongated thin structures), but also helps to alleviate the imbalanced thin and non-thin regions issue [22].

4.1 Generic Object Segmentation (GOS) Network

As our Slim Scissors is model-agnostic to GOS, it can be easily integrated with any generic interactive segmentation network to formulate an end-to-end trainable segmentation framework. In this work, we show an example instantiation based on Inside-Outside Guidance (IOG) [48], the current best performing interactive segmentation algorithm. In summary, IOG takes an inside (object's center) and two outside points (two symmetrical corners of bounding box enclosing the object) as input, outputting an object-centric crop based on the relaxed box. Concatenated with the foreground and background localization heatmaps derived from the inside and outside guidance, the cropped input is resized to 512×512 before passing to a coarse-to-fine network for segmentation. As a coarse-grained prediction usually suffices for segmenting the object's main body, we replace the ResNet-101 [14] backbone in IOG with a much lighter ResNet-18 [14] to reduce computational cost.

4.2 Thin Parts Refinement (TPR) Network

TPR provides a novel solution to segment thin parts by taking advantages of synthetic background, which consists of the following two stages during training.

[5] Beetle's legs and antennas, heron's neck and legs and morpho's antennas.

Scribbles generation. We ask the users to roughly draw some scribbles to identify the thin parts for segmentation. However, in practice, different users tend to have different scribbling behaviors, it is therefore almost infeasible to collect many interaction samples from real users for training. To simulate scribbles annotated by human annotators, we first extract the elongated thin structures for each object instance following [23]. During training, depending on the target object size, these thin regions are randomly dilated by $[0.5r,\ 1.5r]$ pixels to simulate coarse scribbling behavior (definition of r can be found in Eqn. 6.2). The effects of dilations (coarseness of scribbles) will be studied in Sec. 6.3.

Background synthesis. We propose to construct a synthetic background image based on the user-provided scribbles to simulate the absence of thin parts. In particular, we employ an image inpainting algorithm to inpaint the regions indicated by the scribbles. Since scribbles used to mark thin structures typically cover relatively small areas and are often with rich background context, a realistic-looking completed image can be easily obtained given existing inpainting algorithms. In this work, we opt for simplicity and use `cv2.inpaint` [6] from OpenCV [33]. Note that our proposed synthetic background idea is a universal one and more advanced inpainting methods such as [28,46] can be employed but it is out of the scope of this work. We will leave this to our future work. Fig. 3 depicts some examples of generated synthetic background images where elongated thin structures are successfully removed.

Based on this, we concatenate the input image, its corresponding synthetic background and the scribbles mask as input for training the TPR network. Considering an RGB image I, the corresponding synthetic background B and scribbles mask S, the task of segmenting thin structures can be mathematically formulated as learning a mapping function $f(;\theta_{\text{thin}})$ that is parameterized by θ_{thin}:

$$M_{\text{thin}} = f(I, B, S; \theta_{\text{thin}}) \tag{1}$$

where M_{thin} denotes the mask of thin parts.

Network architecture. Similarly, our TPR is realized with a coarse-to-fine network structure. More specifically, taking the cropped input from GOS subnetwork as input, the first stage adopts a ResNet-18 [14]-based encoder-decoder structure, which appends an Atrous Spatial Pyramid Pooling (ASPP) module [6] at the end of the encoder to incorporate global context, followed by a decoder that sequentially upsamples and concatenates with low-level feature for subsequent convolution, producing a coarse mask and features of 32 channels.

Taking the concatenation of the original image, synthesized background, scribbles input, coarse mask prediction and features from the previous stage as input, the second stage first downsamples the concatenated inputs to $0.5\times$ resolution to enlarge the receptive field for refinement, followed by two lightweight convolution layers. The output features is then bilinearly upsampled back to original resolution, concatenated with the (image, background, scribble) input

[6] `https://docs.opencv.org/master/d7/d8b/group__photo__inpaint.html`

before passing to another two convolution layers, followed by a sigmoid activation to produce M_{thin}. Please refer to our supplement for more details.

Training of Slim Scissors. We employ the same training strategy adopted in IOG [48] for training our GOS network, *i.e.* training with binary cross-entropy loss and applying side losses at each level of CoarseNet as deep supervision. Similarly, our TPR network (both decoder and refiner) is trained with standard binary cross-entropy loss, with thin structures extracted from [22] being the ground truth masks. In order to encourage the TPR network to mainly focus on refining the coarse predictions along elongated thin parts, we only consider those background pixels around the thin parts during training. In our experiments, the ratio of foreground and background pixels is set as 1:2.

5 Fast Slim Scissors

As explained in Sec. 3, coarse scribbling provides a more user-friendly way to advance the interactive thin parts segmentation process. Nevertheless, drawing scribbles for each and every thin part can still be time-consuming and redundant since most thin structures share similar appearance or texture (*e.g.*, bug legs). This observation motivates us to answer the following question: *in the scenario where many thin parts are with similar patterns, do users really need to brush them one by one?* The answer is **no**.

Similar detection module. To reduce the annotation burden in scribbling similar thin parts over and over again, we propose a similar detection module (SDM) to augment our Slim Scissors. SDM accepts one or two scribbles from the user and automatically mines other similar thin structures. The newly mined thin structures are often coarse yet sufficient to serve as new scribbles for guiding the subsequent background synthesis in Slim Scissors. In particular, given some user-marked scribbles, we first extract an initial segmentation of thin parts covered by those scribbles using our Slim Scissors, which help separate the scribbled regions into a set of foreground and background pixels. We next compute two distance maps by matching each pixel within the same image to both the foreground F and background sets B in the embedding feature space. Mathematically, for each pixel i, we define its matching as:

$$G_F(i) = \min_{j \in F} D(i,j) \tag{2}$$

$$G_B(i) = \min_{j \in B} D(i,j) \tag{3}$$

where $D(i,j)$ refers to the distance between pixel i and j in term of their corresponding embedding vector e_i and e_j, which is defined as [40,45]:

$$D(i,j) = \begin{cases} 1 - \frac{1}{1+\exp(\|e_i - e_j\|^2 + b_F)}, & \text{if } j \in F \\ 1 - \frac{1}{1+\exp(\|e_i - e_j\|^2 + b_B)}, & \text{if } j \in B \end{cases} \tag{4}$$

where b_F and b_B denote the learnable foreground and background bias.

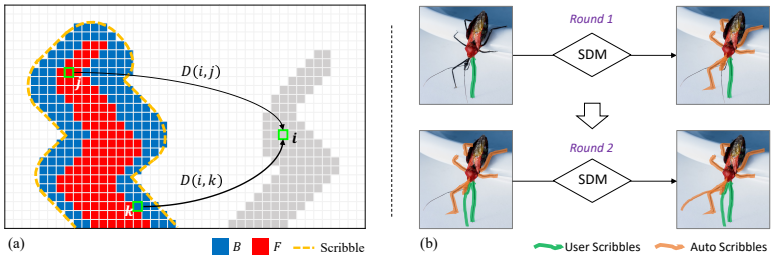


Fig. 4: Scribble generation process using our SDM.

Given the computed foreground and background matching maps, we concatenate them with the pixel-level embedding feature and pass to a lightweight decoder module for outputting a binary mask predicting other similar thin structures to the one(s) highlighted by current scribble(s). As a result, our Fast Slim Scissors only requires one or two scribbles from the user while attaining results similar to that of drawing scribbles on all the elongated thin structures.

In Fig. 4, we show how to apply our SDM to automatically generate more scribbles covering other similar thin parts. Specifically, in round one, the user first draws a coarse scribble (green scribble) on one of the beetle's leg (the most frequent thin parts). Our SDM then identifies the remaining legs and automatically generates coarse scribbles for them (orange scribbles). Similarly, in round two, the user draws another scribble on one of the missing parts, *i.e.*, beetle's antennas, and uses SDM to locate another missing antennas within the image. In practice, such an iterative process will continue until all thin parts are covered by scribbles (either user drawn or generated by our SDM).

Overall process. The overall interactive segmentation process of *Fast* Slim Scissors is summarized as follows: 1) The user first provides an inside (object's center) and two outside clicks (two symmetrical corners of a bounding box) [48] to indicate the object-to-segment; 2) Draw a coarse scribble on any of the thin parts; 3) SDM mines other similar thin structures by computing a global matching with the user-provided scribble(s) (Sec. 5); 4) Along with the previously drawn ones, these newly mined scribbles are passed to our Slim Scissors, producing an initial object segmentation mask including elongated thin structures (Sec. 4); 5) The user could either introduce more scribbles on the missing thin parts or remove the mislabeled thin regions; 6) Step 3 to 5 (step 3 is skipped if the user chose to erase the wrongly segmented thin parts) are repeated until the user is satisfied with the final segmentation output.

6 Experiment

6.1 Datasets and Settings

Implementation details. We train our Slim Scissors on ThinObject-5K `train` split [22] with 4,743 images for 30 epochs. All networks are initialized from

ResNet-18 [14] pre-trained on ImageNet [36]. We train our model using Adam optimizer [18] with batch size of 25 and weight decay of 10^{-4}. The base learning rate of the pre-trained and newly initialized weights are set to 10^{-4} and 10^{-3}, respectively. We employ a linear decay learning rate scheduler with 5 epochs of linear warmup. We randomly resize and rotate images with a scale factor from 0.75 to 1.25 and rotation angle from -20° to 20°. We also apply random horizontal flipping during training. For similar detection experiments, we created a subset from ThinObject-5K `train` split, where each image fulfills the following criteria: 1) each image must contain at least two separated thin regions; 2) the thin regions should belong to the same semantic category (*e.g.*, bug's legs and antennas are considered two different categories). The resulting subset contains 2,128 images. We train our SDM on this subset for 45 epochs with batch size of 16 using a constant learning rate. All other hyperparameters remain the same.

Datasets. We evaluate on the following three benchmarks: 1) **ThinObject-5K**, a large-scale synthetic dataset from [22] that is composed of different objects with elongated thin structures (*e.g.*, ants, racket, harps *etc.*). We use 500 testing images from this dataset for testing; 2) **COIFT** containing 280 natural images of birds and bugs from [31,32]; 3) **HRSOD**. We follow [22] by using a subset of 280 images (305 instances) from HRSOD [47] for evaluation.

Evaluation metrics. We employ the same evaluation metrics as in [22] for evaluation, including standard Intersection-over-Union (IoU), IoU_{thin} for measuring IoU only on regions surrounding the thin structures, and boundary measure $\mathcal{F}$ from [34] for evaluating the quality of segmented edges.

6.2 Comparison with the State-of-the-Art Methods

We first compare our Slim Scissors with prior methods on ThinObject-5K `test` split, COIFT [31] and HRSOD [47] dataset. For our Slim Scissors, we simulate user-drawn scribbles by performing morphological dilation on the thin parts extracted from [22], where kernel radius r is adaptively determined based on the image resolution. The underlying idea is that thin parts in a higher resolution image appear to be thicker, requiring sufficiently coarse scribbles for annotation. Specifically, let the range of kernel radius be $[r_{\min}, r_{\max}]$, the maximum target object size in the three benchmarks be $N_{\max}$, given a test object with pixel number N, the kernel radius is set to

$$r = \left\lfloor r_{\min} + N \times \frac{r_{\max}}{N_{\max}} \right\rfloor, \tag{5}$$

where $r_{\min} = 3$ and $r_{\max} = 240$. The results are summarized in Tab. 1.

We first notice a significant performance gap between the synthetic and real image datasets for the state-of-the-art TOS-Net [22], suggesting that TOS-Net has possbily overfitted on the artefacts present in synthetic image. On the other hand, the performance gap of Slim Scissors is much smaller. More importantly, when tested on real image benchmarks, our Slim Scissors outperforms TOS-Net by a significant margin (4.4% IoU_{thin} and 1.7% $\mathcal{F}$ score on COIFT and 5.9%

Method	Training Set	ThinObject-5K [22]			COIFT [31]			HRSOD [47]		
		IoU	IoU_{thin}	$\mathcal{F}$	IoU	IoU_{thin}	$\mathcal{F}$	IoU	IoU_{thin}	$\mathcal{F}$
DEXTR [22]	PASCAL-10K	61.8	43.5	49.0	70.6	36.8	74.4	69.5	35.4	66.1
DEXTR [22]	ThinObject-5K	88.8	74.0	89.3	88.2	68.3	93.4	82.5	57.7	84.8
f-BRS [38]	ThinObject-5K	90.9	80.6	89.4	87.8	63.6	89.1	78.1	49.7	74.1
f-BRS† [38]	ThinObject-5K	92.5	84.0	92.6	89.9	67.1	94.6	86.4	63.8	87.7
TOS-Net [22]	ThinObject-5K	94.3	86.5	94.8	92.0	76.4	95.3	86.4	65.1	87.9
Slim Scissors (ours)	ThinObject-5K	91.1	80.5	93.5	93.2	80.8	97.0	87.7	71.0	91.4

Table 1: **Quantitative results** on ThinObject-5K `test` set, COIFT and HRSOD dataset. † take scribble and synthetic background as input.

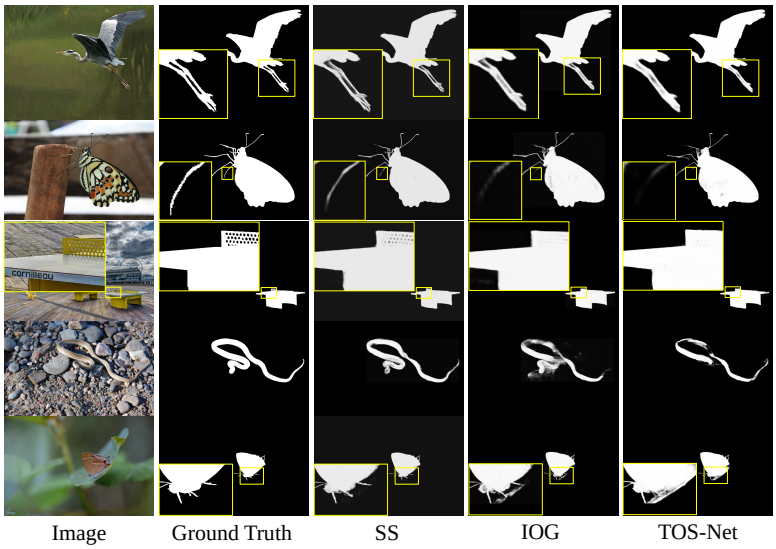


Fig. 5: **Qualitative comparison** between IOG [48], TOS-Net [22] and our SS.

IoU_{thin} and 3.5% $\mathcal{F}$ on HRSOD dataset), demonstrating the better generalization capabilities of our Slim Scissors for practical application. Some qualitative comparison can be found in Fig. 5.

6.3 Ablation Study

We perform ablation experiments to quantitatively justify the effectiveness of each proposed component. The results are summarized in Table 2.

Synthetic background and scribble input. To study the effectiveness of synthetic background for segmentation of thin structures, we construct two baselines that take only image *or* image with synthetic background as input for TPR. As shown in Tab. 2 (top), the performance is significantly boosted when using synthetic background, and can be further improved when using coarse scribbles.

Dual path architecture. As shown in Tab. 2 (bottom), removing TPR (3rd row) reduces to IOG [48] except for using lighter ResNet-18 and being trained on ThinObject-5K dataset, where the performance severely degrades. However, as the large performance gap is partly due to different network input, we also train a baseline that additionally accepts synthetic background and scribbles input (2nd row) for fair comparison. Dual path design performs the best, suggesting explicitly separating the processing of thin parts from the object's main body is generally beneficial for this task.

TPR input	COIFT [31]			HRSOD [47]		
	IoU	IoU_{thin}	$\mathcal{F}$	IoU	IoU_{thin}	$\mathcal{F}$
+BG +scribble	93.2	80.8	97.0	87.7	71.0	91.4
+BG	92.9	79.0	96.7	86.6	68.6	90.3
image only	91.2	76.1	94.3	83.2	60.5	85.0

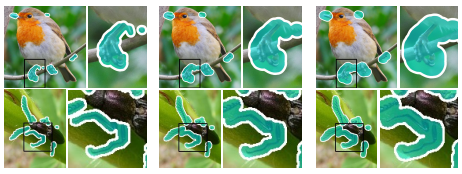
Dual Branch	BG + Scribble	ThinObject-5K [22]			COIFT [31]		
		IoU	IoU_{thin}	$\mathcal{F}$	IoU	IoU_{thin}	$\mathcal{F}$
✓	✓	91.1	80.5	93.5	93.2	80.8	97.0
	✓	91.0	78.7	93.5	93.5	77.9	98.4
		89.0	75.7	88.9	91.3	73.1	94.4

Table 2: Ablation study. **TPR input** and **dual path structure**.

Robustness to scribble size. Lastly, we study the robustness of our Slim Scissors to different scribble size. Thicker scribbles enable faster scribbling but may lead to poorer inpainting result which subsequently affect the segmentation quality; thinner scribbles encode stronger spatial prior but at the cost of being slower. As shown in Fig. 6, as expected, the segmentation performance gradually degrades when using larger r. Nevertheless, we can see that our approach is fairly robust on overall. Even with $1.5\times r$ scheme, our Slim Scissors still consistently outperforms TOS-Net on both benchmarks, well demonstrating its practicality with various scribble size. We also visualize some examples in Fig. 6, where we can see $1\times r$ is sufficient to enable users for fast scribbling over thin parts.

6.4 Automated Scribble Recommendation

To validate the effectiveness of the similar detection module (SDM) in boosting the annotation process, we perform additional experiments to evaluate our fast Slim Scissors. The goal of SDM is to automatically acquire similar thin parts with the given scribbles. We perform evaluation on a subset of COIFT, and choose the objects (*e.g.* birds and bugs) that are with at least two or more similar thin parts, resulting in 268 images and 2063 independent thin parts in total.



(a) $0.5r$ (b) r (c) $1.5r$

r	COIFT [31]			HRSOD [47]		
	IoU	IoU_{thin}	$\mathcal{F}$	IoU	IoU_{thin}	$\mathcal{F}$
$0.5\times$	93.2	81.4	96.9	87.6	71.7	91.3
$1\times$	93.2	80.8	97.0	87.7	71.0	91.4
$1.5\times$	93.0	80.1	96.9	87.4	69.8	91.0

Fig. 6: **Robustness against different scribble coarseness**. r denotes the radius of morphological kernel used to dilate the scribbles for evaluation (Eqn. 6.2).

At the beginning, for each image, we randomly select a thin part and draw a scribble to cover it. SDM is then used to detect the remaining ones. In the following rounds, we progressively conduct additional interactions (*e.g.*, adding new scribbles to the missing parts or removing the false regions detected by SDM) until all the thin parts are well covered by scribbles. As shown in Fig. 7, our fast Slim Scissors reduces human efforts by more than a half (∼1000 scribbles) and achieves competitive performance to that of a fully annotated counterpart (79.9%+ *vs.* 81.7%).

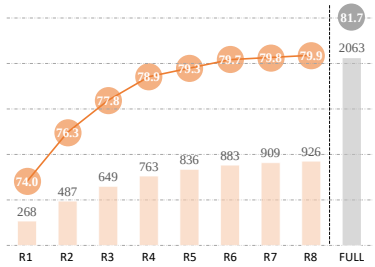
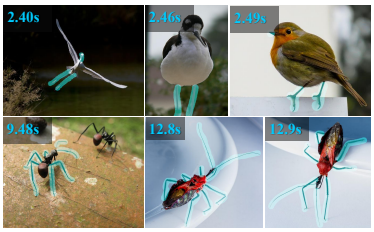


Fig. 7: The trend of IoU_{thin} and the corresponding annotation burden of Fast Slim Scissors on a subset of COIFT.

6.5 User Study

Impact of synthetic scribbles. We conducted a user study to examine the impact of synthetic scribbles. Specifically, we recruited 9 participants and split them into 3 groups where each group is asked to annotate the entire COIFT dataset (280 images) with their best efficiency. In other words, each image is annotated 3 times and annotators can choose their preferred scribble sizes. As shown in Fig. 8, the performance on human-drawn scribbles are close to the synthetic ones, demonstrating the robustness of our approach to user variance.

Compatibility with real user input. We also conducted another user study to examine the effectiveness of the proposed approach with real human user input. We collected 20 images from COIFT dataset[31] and manually chose two sets of scribble thickness for each test image, simulating thin and thick scribbles. Among these 20 images, there are 7 birds, 7 butterflies and 6 others (mainly insects). Some hardest and easiest samples are shown in Fig. 8. We recruited 5 participants where each participant is asked to draw scribbles along the thin parts under 4 different settings: (1) use only the pre-defined thin scribble; (2) use only the pre-defined thick scribble; (3) use thin scribbles with the help of SDM



COIFT	Synthetic		Human-drawn			
	SS	TOS-Net	Group1	Group2	Group3	Average
IoU	93.2	92.0	92.5	92.5	92.5	92.5 ± 0.00
IoU_{thin}	80.8	76.4	78.8	78.8	79.1	78.9 ± 0.14
$\mathcal{F}$	97.0	96.2	96.2	96.3	96.4	96.3 ± 0.08

Fig. 8: **User study.** Left: Top 3 easiest (top) and hardest (bottom) samples with their annotation time. Right: user study performance on COIFT dataset.

Annotation rule	User1		User2		User3		User4		User5		Average	
	IoU_{thin}	Time	IoU_{thin}	Time	IoU_{thin}	Time	IoU_{thin}	Time	IoU_{thin}	Time	IoU_{thin}	Time
Thin scribbles	73.3	178	73.9	219	75.7	218	75.5	251	75.7	210	74.8	215
Thick scribbles	72.4	112	72.5	143	74.1	108	74.0	149	73.8	86.3	73.4	120
Thin scribbles + SDM	73.8	65.2	74.5	76.2	75.6	90.5	75.5	73.6	76.0	59.2	75.1	72.9
Thick scribbles + SDM	72.9	55.7	73.8	51.4	72.7	44.8	74.0	53.4	74.6	33.2	73.6	47.7

Table 3: **User study.** Total time (s) spent on scribbling all the thin regions and IoU_{thin} in 20 images using different annotation rules.

and lastly (4) use thick scribbles with SDM. The total time (seconds) spent on scribbling all the thin regions and IoU_{thin} are reported in Tab. 3. We also include the performance using synthetic scribbles (dilation rate of $1\times r$) as reference.

We first notice that, compared to thin scribbles, employing thick scribbles reduces the annotation time by nearly half with only slight performance degradation. With the help of SDM, the annotation time is further reduced while maintaining similar accuracy. In practice, when SDM is employed, much fewer scribbles are needed, we can therefore afford to use thinner scribbles for better accuracy. Lastly, the small performance gap between synthetic and real human-drawn scribbles demonstrates the robustness of our method to user variance.

7 Conclusion

This work addresses the task of interactively segmenting objects with elongated thin structures, such as bicycles with thin spokes. The underlying core idea is to synthesize a background image without thin parts such that the network could learn to compare the (image, background) pair for extraction of thin structures. Inspired by this, we devise an effective interactive thin object segmentation technique called Slim Scissors that only requires *coarsely*-drawn scribbles covering thin parts, which is arguably easier than precise clicking used in previous literature. We also show that it can be seamlessly combined with existing state-of-the-art interactive segmentation networks to form an end-to-end trainable segmentation framework. To reduce the annotation burden, we also propose a fast Slim Scissors variant, which augments Slim Scissors with a similar detection module for mining additional scribbles covering other similar thin structures in the same image. We demonstrate the effectiveness of our approach on three publicly available benchmarks and achieve the new state-of-the-art.

Limitations. Our proposed SDM occasionally falsely detects irrelevant thin parts (*e.g.*, another bird's legs) in the presence of multiple objects. How to reduce the false positives will be investigated in our future work.

Acknowledgments. This work was supported in part by the National Key R&D Program of China (No.2021ZD0112100), the National NSF of China (No.U1936212, No.62120106009, No.61972405), the Fundamental Research Funds for the Central Universities (No. K22RC00010).

References

1. Acuna, D., Ling, H., Kar, A., Fidler, S.: Efficient interactive annotation of segmentation datasets with Polygon-RNN++. In: CVPR (2018)
2. Bai, X., Sapiro, G.: A geodesic framework for fast interactive image and video segmentation and matting. In: ICCV (2007)
3. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In: ICCV (2001)
4. Castrejon, L., Kundu, K., Urtasun, R., Fidler, S.: Annotating object instances with a Polygon-RNN. In: CVPR (2017)
5. Chen, B., Ling, H., Zeng, X., Gao, J., Xu, Z., Fidler, S.: Scribblebox: Interactive annotation framework for video object segmentation. In: ECCV (2020)
6. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. TPAMI (2018)
7. Chen, X., Zhao, Z., Yu, F., Zhang, Y., Duan, M.: Conditional diffusion for interactive segmentation. In: ICCV (2021)
8. Dang, V.N., Galati, F., Cortese, R., Giacomo, G.D., Marconetto, V., Mathur, P., Lekadir, K., Lorenzi, M., Prados, F., Zuluaga, M.A.: Vessel-captcha: an efficient learning framework for vessel annotation and segmentation. In: Medical Image Analysis (2021)
9. Dong, X., Shen, J., Shao, L., Van Gool, L.: Sub-markov random walk for image segmentation. TIP (2015)
10. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV (2010)
11. Grady, L.: Random walks for image segmentation. TPAMI (2006)
12. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: CVPR (2010)
13. Hao, Y., Liu, Y., Wu, Z., Han, L., Chen, Y., Chen, G., Chu, L., Tang, S., Yu, Z., Chen, Z., et al.: Edgeflow: Achieving practical interactive segmentation with edge-guided flow (2021)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
15. Hu, Y., Soltoggio, A., Lock, R., Carter, S.: A fully convolutional two-stream fusion network for interactive image segmentation. Neural Networks (2019)
16. Jang, W.D., Kim, C.S.: Interactive image segmentation via backpropagating refinement scheme. In: CVPR (2019)
17. Jegelka, S., Bilmes, J.: Cooperative cuts for image segmentation. Tech. rep., Technical Report 2010-0003, University of Washington (2010)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2015)
19. Kontogianni, T., Gygli, M., Uijlings, J., Ferrari, V.: Continuous adaptation for interactive object segmentation by learning from corrections. In: ECCV (2020)
20. Le, H., Mai, L., Price, B., Cohen, S., Jin, H., Liu, F.: Interactive boundary prediction for object selection. In: ECCV (2018)
21. Li, Z., Chen, Q., Koltun, V.: Interactive image segmentation with latent diversity. In: CVPR (2018)
22. Liew, J.H., Cohen, S., Price, B., Mai, L., Feng, J.: Deep interactive thin object selection. In: WACV (2021)
23. Liew, J.H., Cohen, S., Price, B., Mai, L., Ong, S.H., Feng, J.: MultiSeg: Semantically meaningful, scale-diverse segmentations from minimal user input. In: ICCV (2019)

24. Liew, J.H., Wei, Y., Xiong, W., Ong, S.H., Feng, J.: Regional interactive image segmentation networks. In: ICCV (2017)
25. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV (2014)
26. Lin, Z., Zhang, Z., Chen, L.Z., Cheng, M.M., Lu, S.P.: Interactive image segmentation with first click attention. In: CVPR (2020)
27. Ling, H., Gao, J., Kar, A., Chen, W., Fidler, S.: Fast interactive object annotation with Curve-GCN. In: CVPR (2019)
28. Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B.: Image inpainting for irregular holes using partial convolutions. In: ECCV (2018)
29. Majumder, S., Yao, A.: Content-aware multi-level guidance for interactive instance segmentation. In: CVPR (2019)
30. Maninis, K.K., Caelles, S., Pont-Tuset, J., Van Gool, L.: Deep extreme cut: From extreme points to object segmentation. In: CVPR (2018)
31. Mansilla, L.A., Miranda, P.A.: Oriented image foresting transform segmentation: Connectivity constraints with adjustable width. In: SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI) (2016)
32. Mansilla, L.A., Miranda, P.A., Cappabianco, F.A.: Oriented image foresting transform segmentation with connectivity constraints. In: ICIP (2016)
33. OpenCV: Open source computer vision library (2015)
34. Perazzi, F., Pont-Tuset, J., McWilliams, B., Gool, L.V., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: CVPR (2016)
35. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: ACM ToG (2004)
36. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
37. Sofiiuk, K., Petrov, I., Barinova, O., Konushin, A.: f-BRS: Rethinking backpropagating refinement for interactive segmentation. In: CVPR (2020)
38. Sofiiuk, K., Petrov, I., Barinova, O., Konushin, A.: f-brs: Rethinking backpropagating refinement for interactive segmentation. In: CVPR (2020)
39. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: CVPR (2008)
40. Voigtlaender, P., Chai, Y., Schroff, F., Adam, H., Leibe, B., Chen, L.C.: FEELVOS: Fast end-to-end embedding learning for video object segmentation. In: CVPR (2019)
41. Wang, G., Zuluaga, M.A., Li, W., Pratt, R., Patel, P.A., Aertsen, M., Doel, T., Divid, A.L., Deprest, J., Ourselin, S., et al.: DeepIGeoS: a deep interactive geodesic framework for medical image segmentation. TPAMI (2018)
42. Wu, J., Zhao, Y., Zhu, J.Y., Luo, S., Tu, Z.: Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In: CVPR (2014)
43. Xu, N., Price, B., Cohen, S., Yang, J., Huang, T.: Deep GrabCut for object selection. In: BMVC (2017)
44. Xu, N., Price, B., Cohen, S., Yang, J., Huang, T.S.: Deep interactive object selection. In: CVPR (2016)
45. Yang, Z., Wei, Y., Yang, Y.: Collaborative video object segmentation by foreground-background integration. In: ECCV (2020)
46. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: ICCV. pp. 4471–4480 (2019)

47. Zeng, Y., Zhang, P., Zhang, J., Lin, Z., Lu, H.: Towards high-resolution salient object detection. In: ICCV (2019)
48. Zhang, S., Liew, J.H., Wei, Y., Wei, S., Zhao, Y.: Interactive object segmentation with inside-outside guidance. In: CVPR (2020)